Détection de communautés dans les flots de liens par optimisation de la modularité

Emmanuel Orsini

Détection de communautés dans les flots de liens par optimisation de la modularité

Emmanuel Orsini

Sorbonne Université, UPMC Univ Paris 06, UMR 7606, LIP6, F-75005, Paris, France
CNRS, UMR 7606, LIP6, F-75005, Paris, France
emmanuel.orsini@polytechnique.edu

RÉSUMÉ. *L'article qui suit propose de donner un sens à la modularité dans les flots de liens et ainsi de bénéficier de certaines de ses propriétés, et des heuristiques qui l'optimisent. Cette notion de modularité aboutira après quelques simplifications à un algorithme capable de calculer une partition sur un jeu de données de 400 000 emails. Pour ce faire on construira une nouvelle modélisation où le temps est complètement continu, sur laquelle la modularité se définit naturellement et de manière pertinente. Cette modélisation apporte une nouvelle interpretation des réseaux dynamiques, qui se veut suffisamment générale pour s'adapter à différents types de données.*

ABSTRACT : *The following paper aims at defining modularity in link streams in order to take advantage of its properties and of the heuristics that optimize it. This concept of modularity leads after a few simplifications to an algorithm able to compute a partition of a 400,000 emails data set. To reach this goal we will build a new model, in which time is completely continuous and modularity is defined in a natural and relevant way. This model results in a new interpretation of dynamic networks, broad enough to adapt to different types of data.*

MOTS-CLÉS : *Flot de liens, Communautés, Modularité, Réseau dynamique*

KEYWORDS : *Link streams, Communities, Modularity, Dynamic network*

soumission à *MARAMI 2015*, le 6 septembre 2015.

1. Introduction

La structure des graphes réels (ou graphes de terrain) diffère très clairement de celle des graphes aléatoires : Un graphe de terrain se distingue en particulier par l'existence de groupes de nœuds fortement interconnectés et faiblement connectés avec l'extérieur, que l'on appelle les communautés. Cette structure a de nombreuses conséquences et résulte souvent de vérités de terrain (groupes d'amis dans un réseau social, proximité géographique, etc). Il est donc intéressant de définir et de détecter automatiquement ces communautés, et la littérature sur ce sujet est déjà riche [3]. L'approche classique consiste à définir une fonction de qualité qui attribue une note à chaque partition. Le problème revient ensuite à chercher les heuristiques qui optimisent le mieux cette fonction de qualité. En particulier la modularité [4], dont les principes seront rappelés dans la partie suivante, est l'une des fonctions de qualité qui fait le plus autorité. La méthode de Louvain [2] est à ce jour l'heuristique la plus utilisée car elle est rapide et produit de très bons résultats.

Pour étudier la dynamique de communautés, certains travaux [1] détectent de communautés sur chaque « snapshot » du graphe, qui sont l'état du graphe à un instant donné. Ils cherchent ensuite à observer des fusions, fissions, apparitions ou disparitions de communautés entre deux snapshots successifs. On explorera ici une approche très différente puisque l'on étudiera la structure communautaire sur un modèle intrinsèquement temporel : le flot de liens [6, 7]. Un flot de liens, comme un graphe, se définit par un ensemble de nœuds V et un ensemble de liens E, mais avec $E \subset \mathbb{R} \times V \times V$, le réel désignant un instant. Ainsi le lien $(t, a, b) \in E$ siginifie que les nœuds a et b interagissent à l'instant t. Dans ce cadre les communautés sont un groupe de nœuds qui interagissent beaucoup entre eux sur une période de temps donnée.

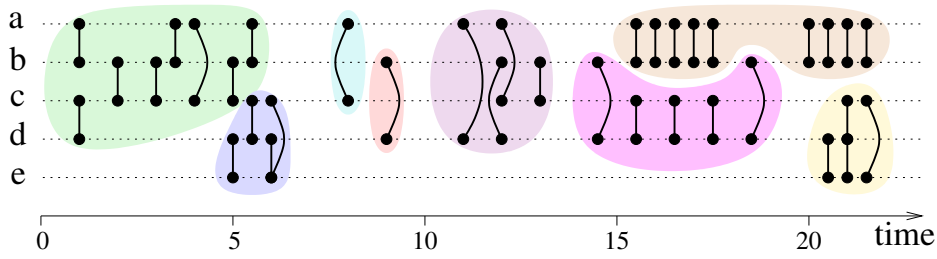

Figure 1. *Une représentation d'un flot de liens à 5 nœuds. Les zones colorées sont les communautés. Les trois premiers liens du flot sont* $(1.0, a, b)$, $(1.0, d, e)$ *et* $(2.0, b, c)$

Aucun découpage du temps en snapshots ne parait adapté à la détection de communautés sur l'exemple qui précède. En effet les flots de liens font une meilleure approche sur des données de type "interactions", alors que les snapshots conviennent à l'analyse de "relations". Les interactions sont intrinsèquement datées, comme par exemple un envoi de mail ou une poignée de main. Les relations au contraire durent dans le temps, comme les relations d'amitié. Les flots de liens permettent donc de trai-

ter des données d'une autre nature, et ainsi d'observer des communautés d'une autre nature.

2. La modularité dans les graphes

Considérons un graphe $G = (V, E), |V| = n, |E| = m$ et une partition des nœuds $C_1, \ldots, C_K$. Soit $A : V^2 \to \{0, 1\}$ la matrice d'adjacence associée à G. On appelle e_{ij} la proportion des arêtes du graphe qui sont entre C_i et C_j.

$$e_{ij} = \frac{|(C_i \times C_j) \cap E|}{m} = \sum_{u,v \in V^2} \frac{A(u,v)}{2m} \mathbf{1}_{u \in C_i} \mathbf{1}_{v \in C_j} \tag{1}$$

On note par ailleurs $a_i = \sum_{j=1}^{K} e_{ij} = \sum_{u,v \in V^2} \frac{A(u,v)}{2m} \mathbf{1}_{u \in C_i}$ la proportion de « demi-arêtes », ou « proportion de degré » dans C_i.

Remarque : En particulier e_{ii} est la proportion d'arêtes internes à C_i et l'on a $\sum_{i=1}^{K} e_{ii} \leqslant 1$

Remarque : On a également $\sum_{i=1}^{K} a_i = \sum_{i=1}^{K} \sum_{j=1}^{K} e_{ij} = 1$

La modularité $Q \in]-1, 1[$ de cette partition du graphe G est définie comme :

$$Q = \sum_{i=1}^{K} (e_{ii} - a_i^2) \tag{2}$$

On peut l'interpréter comme la proportion des arêtes internes aux communautés moins l'espérance de cette proportion si l'on réoriente aléatoirement les arêtes. Une partition qui rend bien compte de la structure communautaire de G aura une forte modularité alors qu'une partition aléatoire aura une modularité proche de zéro.

3. Modularité dans les flots de liens

Le modèle du flot de liens n'est pas complètement continu dans le sens où un nombre fini de dates le caractérise : les dates des interactions. Cette partie s'attachera à construire un modèle qui fait intervenir tous les instants du temps pour tous les nœuds. On manipule donc un ensemble de « nœuds datés » et une fonction d'adjacence qui les lie entre eux.

Convention 1 On parlera simplement de « nœud » ou de « nœud du flot de liens » pour désigner un élément u de $V = \{1, \ldots, n\}$, par opposition au « nœud daté » qui

désigne un élément (u, t_u) de $\tilde{V} = V \times T = \{1, \ldots, n\} \times \mathbb{R}$ qui se lit « le nœud u à l'instant t_u ».

Pour coller à cette approche continue, les liens auront une incidence sur la fonction d'adjacence non pas à un instant unique, mais sur une période de temps autour de la date du lien. Ceci sera détaillé plus tard.

On fait appel à une fonction d'adjacence sur un ensemble $\tilde{V}$ non dénombrable, ce qui pose quelques problèmes que cette section cherchera à dépasser. La partie 3.1 pose ce qu'est une communauté dans ce graphe, la partie 3.2 propose une manière d'y définir l'adjacence et enfin la partie 3.3 y adapte la modularité.

3.1. *Partitionnement en communauté*

Soit F un flot de liens à n nœuds. Alors une communauté est une partie C de $\tilde{V} = V \times \mathbb{R}$ qui se lit comme suit :

« le nœud u appartient à la communauté C au temps t si et seulement si $(u, t) \in C$ »

Une partition en communauté de F est donc une partition de $\tilde{V} = V \times \mathbb{R}$, l'ensemble des nœuds datés.

3.2. *Fonction d'adjacence*

On souhaite définir l'adjacence $A((u, t_u), (v, t_v))$ entre deux nœuds datés (u, t_u) et (v, t_v) qui mesure combien (u à t_u) et (v à t_v) sont connectés. Les exigences sont les suivantes :

1) Si u et v ne communiquent jamais (n'ont pas de liens entre eux) alors leur adjacence est nulle pour tout t_u et t_v.

2) S'il existe un lien (u, v) à t_0 alors l'adjacence $A((u, t_u), (v, t_v))$ est grande si t_u et t_v sont proches de t_0.

3) Si t_u et t_v sont éloignés alors l'adjacence est faible.

4) Si il n'y a pas de liens entre u et v à un temps proche de t_u ou t_v alors l'adjacence est faible.

Remarque : (u, t_u) et (v, t_v) peuvent avoir une adjacence positive même si $t_u \neq t_v$. Ceci est fondamental pour que les nœuds datés aient une structure communautaire. En effet si l'on n'autorise que des communication à temps égaux alors (u, t_u) et (u, t'_u) avec $t_u \neq t'_u$ n'auront aucun chemin d'adjacence positive qui les relie. Alors formellement ils sont dans des « composantes connexes » distinctes et une fonction de qualité mettra toujours une meilleure note quand ces points déconnectés sont placés dans des communautés différentes, aussi près soient t_u et t_v. Il faut donc autoriser des

liens qui traversent le temps ($t_u \neq t_v$) car on veut que nos communautés aient une existence dans le temps.

Autrement dit, on ne veut pas que la partition qui place chaque lien dans sa communauté obtienne systématiquement la note maximale.

Remarque : Il est nécessaire d'introduire une constante de temps. D'ailleurs les autres approches de détection de communautés dans les flots de liens [8, 5] en introduisent systématiquement une. En effet si deux interactions sont espacées d'un temps Δt, comment décider si les placer dans la même communauté a un sens sans comparer Δt à un temps caractéristique ?

Notons δ la constante de temps. Les quatre points vus plus hauts se reformulent à l'aide de δ ainsi :

1) si u et v ne communiquent jamais alors $\forall (t_u, t_v) \in \mathbb{R}, A((u,t_u),(v,t_v)) = 0$

2) s'il existe un lien (u,v) à t_0 alors l'adjacence $A((u,t_u),(v,t_v))$ est grande si $|t_u - t| \leq \delta$ et $|t_v - t| \leq \delta$

3) $|t_v - t_u| \gg \delta \Rightarrow A((u,t_u),(v,t_v)) \ll 1$

4) si pour toute date d'interaction t entre u et v on a ($|t_v - t| \gg \delta$ ou $|t_v - t| \gg \delta$) alors $A((u,t_u),(v,t_v)) \ll 1$

Pour satisfaire ces quatre exigences on procédera de la manière suivante : pour tout nœud a et $t_0 \in \mathbb{R}$ on définit la fonction $f_{a,t_0,\delta}$ telle que :

$$f_{a,t_0,\delta} : (u \in V, t_u \in \mathbb{R}) \longmapsto \frac{1}{\delta\sqrt{2\pi}} e^{\frac{-1}{2}\left(\frac{t_u - t_0}{\delta}\right)^2} \mathbf{1}_{u=a}$$

Cette fonction doit être interprétée comme une « distribution de poids » ou une « distribution de degré » sur le nœud a au voisinage de t_0. La gaussienne permet de rendre compte du fait que le poids est réparti de façon symétrique et centrée autour de t_0 et sur un écartement caractéristique de taille δ. L'intégrale de la loi normale sur $\mathbb{R}$ vaut 1 ce qui permet de contrôler le poids total distribué. Ceci se rapproche d'une estimation par noyaux gaussiens. Au delà de ces considérations le choix de la gaussienne est arbitraire.

Le lien $k = (t_0, a, b)$ distribue du poids autour de t_0 sur les nœuds a et b comme défini par sa «fonction de distribution» g_k :

$$g_k : (u, t_u) \longmapsto f_{a,t_0,\delta}(u,t_u) + f_{b,t_0,\delta}(u,t_u)$$

La fonction d'adjacence est alors définie par la formule :

$$A((u,t_u),(v,t_v))\,\mathrm{d}t_u\,\mathrm{d}t_v = \sum_{k=1}^{m} g_k(u,t_u) g_k(v,t_v)\,\mathrm{d}t_u\,\mathrm{d}t_v$$

Exemple 1 Voici la représentation de $(t_u,t_v) \to A((u,t_u),(v,t_v))$ avec deux interactions entre u et v, avec $\delta = 1$. Sur le deuxième schéma les interactions sont plus proches dans le temps.

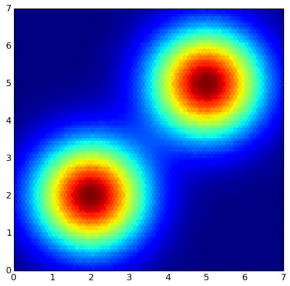

Figure 2. *Interactions à* $t = 2$ *et* 5.

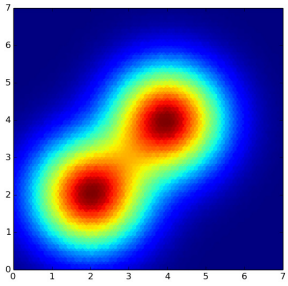

Figure 3. *Interactions à* $t = 2$ *et* 4.

Convention 2 Cette adjacence définit naturellement une notion de degré u d'un nœud à l'instant t_u :

$$\deg(u, t_u) = \sum_{v \in V, v \neq u} \int_{\mathbb{R}} A((u, t_u), (v, t_v)) \, \mathrm{d}t_v = \sum_{k=1}^{m} g_k(u, t_u) \tag{3}$$

La dernière expression, obtenue par interversion somme-intégrale, permet de comprendre pourquoi on peut interpréter g_k comme la « distribution de degré » du nœud k.

Remarque : Le degré instantané vérifie :

$$\int_{\mathbb{R}} \deg(u, t_u) \, \mathrm{d}t_u = \deg(u) \tag{4}$$

Avec $\deg(u)$ le nombre de liens qui touchent u.

Remarque : Un lien $k = (t_0, a, b)$ crée de la distribution de poids entre a et b, entre b et a, mais aussi entre a et lui-même et entre b et lui-même. En effet $g_k(u, t_u) g_k(v, t_v)$ est strictement positif si et seulement si $u \in \{a, b\}$ et $v \in \{a, b\}$, ce qui se produit 4 fois quand on somme sur tous les couples $(u, v) \in V^2$. D'où :

$$\sum_{(u,v) \in V^2} \int_{\mathbb{R}^2} A((u, t_u), (v, t_v)) \, \mathrm{d}t_u \, \mathrm{d}t_v = 4m \tag{5}$$

3.3. *La modularité qui en découle*

Maintenant que les partitions et l'adjacence sont définies, les e_{ij} des flots de liens s'écrivent en adaptant la formule des graphes statiques (1) vue plus haut. Soient $C_1, \ldots, C_K$ les communautés :

$$e_{ij} = \sum_{(u,v) \in V^2} \int_{\mathbb{R}^2} \frac{A((u, t_u), (v, t_v))}{4m} \mathbf{1}_{(u,t_u) \in C_i} \mathbf{1}_{(v,t_v) \in C_j} \, \mathrm{d}t_u \, \mathrm{d}t_v \tag{6}$$

De même pour les a_i :

$$a_i = \sum_{j=1}^{K} e_{ij} = \sum_{u \in V} \int_{\mathbb{R}} \frac{\deg(u, t_u)}{2m} \mathbf{1}_{(u,t_u) \in C_i} \, \mathrm{d}t_u \tag{7}$$

Remarque : Comme sur les graphes statiques, on peut interpréter e_{ij} comme la proportion de l'adjacence totale qui est entre les communautés C_i et C_j ; et a_i comme la proportion du degré total qui est dans la communauté C_i. Le fait que $\tilde{V}$ soit infini ne pose pas de problèmes particuliers, les sommes deviennent seulement des intégrales.

Ainsi la modularité du flot de liens se définit par la même expression (2) que dans les graphes statiques :

$$Q = \sum_i (e_{ii} - a_i^2) \tag{8}$$

Avec ces notations, on est à présent capable de mettre une note à une partition du flot de liens. Cette note n'a été conçue que pour classer les partitions entre elles et non pour donner une indication sur la qualité d'une partition seule ou démontrer la structure communautaire d'un flot de liens. La partie qui suit cherche à trouver une partition ayant la meilleure note possible pour un flot de liens donné et ce dans un temps raisonnable.

Remarque : Contrairement à d'autres modèles, les grandeurs que l'on définit ici sont stables si l'on perturbe les dates des interactions. La matrice d'adjacence est continue par rapport aux dates de liens, ce qui est intéressant car les données réelles sont en partie le fruit de perturbations imprévisibles.

4. Mise en pratique

Optimiser la modularité sur le flot de liens revient donc à l'optimiser sur les nœuds datés $\tilde{V}$ muni de sa fonction d'adjacence. Cet ensemble est infini donc on ne peut pas utiliser les méthodes d'optimisation de la modularité des graphes statique sans simplifications. Cette partie en propose plusieurs qui permettent d'obtenir un partitionnement en communauté sur des gros jeux de données (400 000 mails).

4.1. *Partitionnement des interactions*

Plutôt que de chercher une partition de $\tilde{V} = V \times \mathbb{R}$, on peut se limiter à des structures plus simples. Ce paragraphe a pour but de montrer que l'on peut agréger les nœuds datés de $\tilde{V}$ temporellement pour simplifier le graphe sur lequel on fera le calcul de communautés. Cela revient à faire manuellement une étape de la méthode de Louvain.

Considérons un nœud u du flot de liens. Pour ce nœud on doit faire une partition de $\mathbb{R}$ qui dit à quelle communauté appartient le nœud à chaque $t \in \mathbb{R}$.

Il n'y a pas d'intérêt à changer plusieurs fois un nœud du flot de liens u de communauté entre deux interactions successives de celui-ci. Pour simplifier on suppose que les changements de communautés ne peuvent se produire qu'aux instants exactement au milieu de deux interactions.

Ceci peut se traduire de la manière suivante : soit $t \in \mathbb{R}$, et t' le moment le plus proche de t où u interagit. On impose dorénavant que u soit dans la même communauté à t et t'.

Quelles conséquences sur le graphe que l'on veut construire ?

Notons $t_1, \ldots, t_f$ les instants distincts où u interagit rangés par ordre croissant. On a maintenant découpé $\mathbb{R}$ en f parties de la manière suivante :

$$\left]-\infty, \frac{t_1+t_2}{2}\right[, \left[\frac{t_2+t_3}{2}, \frac{t_3+t_4}{2}\right[, \ldots, \left[\frac{t_{f-1}+t_f}{2}, +\infty\right[$$

Ces f agrégats sont appelés les méta-nœuds de u. Ce découpage peut être fait pour tous les nœuds du flot de liens. Les partitions des méta-nœuds reviennent aux partitions des interactions, ce qui a souvent un sens vis-à-vis des données de terrain.

De la même manière que l'on peut calculer l'adjacence entre deux communautés C_i et C_j par une somme, on peut définir l'adjacence entre deux méta-nœuds M_i et M_j :

$$\sum_{(u,v)\in V^2} \int_{\mathbb{R}^2} A((u,t_u),(v,t_v))\mathbf{1}_{(u,t_u)\in M_i}\mathbf{1}_{(v,t_v)\in M_j}\,\mathrm{d}t_u\,\mathrm{d}t_v$$

Le graphe à construire a maintenant un nombre de nœud borné par le nombre d'interactions, c'est à dire $2m$, ce qui peut suffire à rendre notre algorithme terminable en un temps raisonnable. La partie la plus longue reste néanmoins celle qui construit le graphe des méta-nœuds.

4.2. *Partitionnement des liens*

Pour gagner encore du temps d'exécution, on va restreindre notre recherche aux partitionnements sur les liens. On va imposer que les deux méta-nœuds formés par les deux extrémités d'un même lien soient dans la même communauté. Le graphe qui sert d'entrée à la méthode de Louvain aura déjà fusionnés ces deux méta-nœuds. Cela semble raisonnable car même sans contrainte, cette agrégation se produit naturellement très souvent. Le nombre de nœuds du graphe sera seulement divisé par deux, mais ce n'est pas là que l'on gagne le plus de temps. On autorise en plus la fonction d'adjacence à être « purement temporelle ». Cela signifie que l'on considère qu'il n'y a pas d'adjacence entre différents nœuds du flot de liens :

$$u \neq v \Rightarrow A((u,t_u),(v,t_v)) = 0$$

Remarque : En faisant cela on ne considère plus que la moitié du poids, et l'on a maintenant :

$$\sum_{(u,v)\in V^2} \int_{\mathbb{R}^2} A((u,t_u),(v,t_v))\,\mathrm{d}t_u\,\mathrm{d}t_v = 2m \tag{9}$$

Comme on ne cherchera de connexions qu'entre méta-nœuds qui appartiennent au même nœud du flot de liens, on peut espérer gagner un ordre de grandeur à l'exécution.

Même si elle donne des résultats relativement proche, cette méthode change la fonction d'adjacence et s'éloigne donc dans une certaine mesure de l'idée de base pour se rapprocher d'une méthode de type « line graph » [8]. L'approche line graph construit également un graphe dont les nœuds sont les liens, avec une arête entre deux liens si ils partagent un nœud et que leurs dates sont espacées de moins de Δ (la constante de temps).

4.3. *Troncature de la fonction de distribution*

On a utilisé jusqu'ici comme distribution de poids un fonction gaussienne qui est strictement positive sur chaque réel. Pour rendre la matrice d'adjacence plus creuse, on va négliger le poids déposé à plus de 3δ du centre de la gaussienne. Autrement dit

$$|t_v - t_u| \gg \delta \Rightarrow A((u,t_u),(v,t_v)) \ll 1$$

devient :

$$|t_v - t_u| \gg \delta \Rightarrow A((u,t_u),(v,t_v)) = 0$$

4.4. *Complexité*

La partie la plus lourde est celle qui consiste à déterminer l'adjacence entre les méta-nœuds, on ne mentionnera donc que celle-ci dans le calcul de complexité. Voici une version simplifié de l'algorithme utilisé :

Algorithme 1

Pour chaque point d'interaction p :
  close = l'ensemble des méta-nœuds à une distance inférieure à 3δ du point d'interaction
  Pour chaque méta-nœud a dans *close* :
    Pour chaque méta-nœud b dans *close* :
      Calculer l'adjacence que le point d'interaction p crée entre a et b.

La complexité vaut donc le nombre de points d'interaction multiplié par la taille moyenne de *close* au carré. Il y a alors deux interprétations : on peut estimer que

la taille de *close* est proportionnelle à δ, pour obtenir une complexité moyenne de $O(m\delta^2)$. On peut au contraire considérer que δ est une propriété du flot de liens, et il doit être choisi de telle sorte que *close* fasse en moyenne toujours la même taille indépendamment du flot de liens étudié. Ce raisonnement mène à une complexité moyenne de $O(m)$

5. Expérimentations

5.1. *Problème jouet*

On reprend ici le flot de liens donné en exemple dans l'introduction. Il possède 5 nœuds et 35 liens. Les colorations montrent les différentes communautés détectées avec un δ de 1.

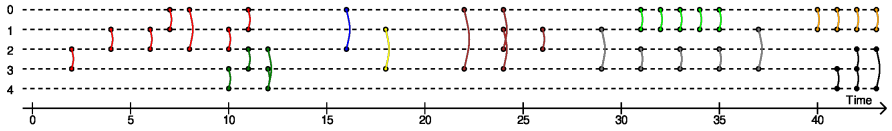

Figure 4. *Communautés détectées sur le problème jouet.*

Remarquons que l'on obtient presque les communautés telles qu'elles sont intuitées sur le schéma de l'introduction. La seule différence est que la communauté verte clair et orange sont séparées ici. En effet il n'y a d'intérêt à les coller que si l'on cherche des communautés qui tiennent compte des schémas périodiques dans le temps, ce qui n'est pas le cas ici.

Une des forces de notre approche est qu'elle est capable de distinguer la communauté rouge de la communauté verte. En effet sur le nœud 2 la dernière interaction de rouge et la première interaction de vert ne sont espacées que d'une unité de temps.

5.2. *Données réelles*

Les données réelles que l'on utilise sont les échanges de mails sur le forum Debian. Le flot de liens comporte 36 097 nœuds pour 393 016 liens. Ces mails sont classés par « threads », qui ont la forme de communautés de liens, car ils sont le résultat de groupes de personnes qui interagissent densément sur une période de temps donnée. On espère donc que notre algorithme puisse détecter cette vérité de terrain. Les liens se répartissent en 128 632 threads, soit environ 3 liens par thread en moyenne. Pour obtenir un partitionnement avec le même nombre de threads, le δ correspondant est de 2 000 secondes, soit un peu plus d'une demi heure. L'algorithme mets environ 30 minutes pour finir. Les résultats obtenus confirment ce que l'on trouve dans [8], c'est à dire que les threads n'ont pas une structure communautaire telle que l'on la définit ici.

Les scores d'information mutuelle normalisée ne peuvent distinguer la partition faite avec notre algorithme de celle faite avec un line graph [8], ni même de la partition qui sépare tous les liens. Les threads des mails Debian ont donc une structure très différente des communautés du problème jouet. En particulier les nœuds sont très inégaux : le nombre d'interactions par utilisateur varie fortement.

6. Limitation

La limite qui suit semble difficile à dépasser dans une approche qui consiste à transformer le flot de liens en un graphe . Elle se retrouve d'ailleurs dans d'autres modèles comme par exemple celui du line graph. En effet le problème vient directement de l'exigence sur l'adjacence énoncée dans la partie 2.2 :

« Si t_u et t_v sont éloignés alors l'adjacence est faible. »

Ceci se traduit par :

$$|t_v - t_u| \gg \delta \Rightarrow A((u, t_u), (v, t_v)) \ll 1$$

Même si cette exigence est raisonnable a priori, elle pose problème dès que l'on cherche à détecter des communautés de taille grandes devant δ. Le début et la fin de la communauté seront nécessairement faiblement interconnectés, et une partition qui coupe cette longue communauté en morceaux aura une meilleure note qu'une partition qui la laisse intacte. Tout se passe comme quand on cherche une partition de modularité maximale sur un graphe statique en forme de fil (avec le nœud i uniquement connecté au nœud $i-1$ et $i+1$). Quand le nombre de nœuds n est grand la partition de modularité maximale est celle qui forme $\sqrt{n}$ communautés de $\sqrt{n}\,\mathrm{nuds}$.

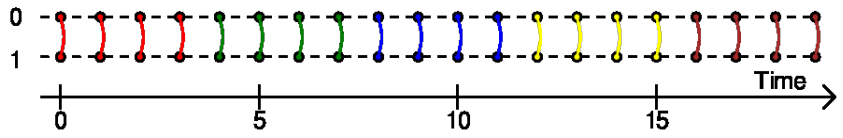

Figure 5. *Une communauté de 20 liens coupée en 5*

Le graphe induit par une longue communauté dans le flot de liens n'a donc pas la structure communautaire que l'on souhaiterait : Il n'est pas assez dense, a un grand diamètre, et a donc tendance à se segmenter quand on cherche les partitions de modularité maximale.

Une possibilité pour dépasser cette limite consiste à redéfinir l'adjacence à la fin de l'algorithme, par exemple en augmentant δ ou en supprimant des liens. On peut alors réitérer l'algorithme pour lui donner l'occasion de fusionner davantage les communautés. Les essais dans cette direction se sont révélés infructueux, tombant dans deux écueils : ne rien changer ou au contraire tout fusionner.

7. Conclusion

Cet article définit une méthode de détection de communautés sur les flots de liens qui aboutit à un algorithme, et il propose par la même occasion un modèle, celui des nœuds datés et de la fonction d'adjacence, dont la généralité permet d'agrandir sans difficulté sa portée. Supposons par exemple que l'on souhaite trouver des communautés dans un flot de liens avec durée. En utilisant comme fonction de distribution une fonction dont l'étalement caractéristique est égal à la durée du lien, on obtiendra toujours une fonction d'adjacence entre les nœuds datés, qui définit ensuite la modularité d'une partition. Le caractère continu du temps dans ce modèle permet de se libérer formellement de la discrétisation, et de se rapprocher du traitement du signal, ce qui peut donner naissance à l'étude de nouveaux objets. La définition du partitionnement en communautés donnée dans la partie 3.1 est volontairement la plus générale possible, pour laisser le soin au lecteur de restreindre cette définition selon sa pertinence face aux vérités de terrain. Pour les flots de liens, on se contente de se restreindre aux partitions sur les interactions, comme présenté dans la partie 4.1.

8. Bibliographie

[1] Aynaud, Thomas and Fleury, Eric and Guillaume, Jean-Loup and Wang, Qinna. Communities in Evolving Networks : Definitions, Detection, and Analysis Techniques. Dans *Dynamics On and Of Complex Networks, Volume 2*, pages 159–200, 2013.

[2] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte et Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics : Theory and Experiment*, 2008(10) :0, 2008.

[3] Santo Fortunato. Community detection in graphs. *Physics Reports*, 486(3) :75–174, 2010.

[4] M. E. J. Newman. From the Cover : Modularity and community structure in networks. *Proceedings of the National Academy of Science*, 103 :8577–8582, jun 2006.

[5] François Queyroi. Delta-Duplication of Time-Varying Graphs. Dans *Conférence MARAMI 2014 (Modèles et Analyses Réseau : Approches Mathématiques et Informatiques)*. PARIS, France, Oct 2014.

[6] Jordan Viard et Matthieu Latapy. Identifying roles in an ip network with temporal and structural density. Dans *Computer Communications Workshops (INFOCOM WKSHPS), 2014 IEEE Conference on*, pages 801–806. IEEE, 2014.

[7] Jordan Viard, Matthieu Latapy et Clémence Magnien. Computing maximal cliques in link streams. *ArXiv preprint arXiv :1502.00993*, , 2015.

[8] Qinna Wang. Link prediction and threads in email networks. *2014 International Conference on Data Science and Advanced Analytics (DSAA2014)*, , 2014.